Fig. 7 is a drop table illustrating superpixel defi-
nition for last-stage expression of a four-bit error-
diffusion system;

Fig. 8 is a like table but showing superpixel defini-
tion in so-called "superpixel families", for four differ-
ent permutations (identified as "0" through "3") of a
four-bit system;

Fig. 9 is a diagram illustrating use of a so-called
"expansion matrix" for conversion from error-diffusion
state to superpixel assignment (in an example converting
from 12 dots/mm and three bits, to 25 dots/mm and two
bits);

Fig. 10 is a table illustrating a four-permutation
superpixel definition (at 25x25 dots/mm);

Fig. 11 is a group of four coordinated graphs, of
which the first (upper) pair of graphs relates halftone
value to "contone" (continuous tone) color tonal level,
for a single-bit binary system and a two-bit system — and
so illustrate a conceptual extrapolation of error diffu-
sion from binary to multibit; and of which the second
(lower) pair of graphs represents the contone functions
themselves — _i. e._, illustrates application of lineariza-
tion curves and thresholds to multilevel error diffusion;

Fig. 12 is a linearization curve for black — _i. e._,
a graph of linearized black vs. contone input, nine and
eight bits per pixel respectively — and particularly
representing a preferred embodiment that is part of a
commercial product;

Fig. 13 is a diagram like Fig. 4 but for the Fig. 1
ink-limiting and plane-split stages (particularly repre-
senting acquisition of the "_factor_" described in the
associated text); and

1   Fig. 14 is a highly schematic diagram showing cyan
2   (C) and magenta (M) separation in the Fig. 13 limiting and
3   split stages.
4
5
6
7   ## DETAILED DESCRIPTION
8   ## OF PREFERRED EMBODIMENTS
9
10
11   1.   APPARATUS-MODULE AND BUSINESS-ENTITY INTERRELATIONS
12
13   Preferred apparatus embodiments of the invention
14   involve three major modules 113, 121E, 141 (Fig. 1), one
15   of which can include an optional internal module 121N.   Of
16   these four units, two are parts of the environment of the
17   invention, not elements of the invention itself as most
18   broadly regarded: a computer 113 and an internal RIP
19   121N.
20   The remaining two units are elements of at least some
21   of the previously introduced major apparatus aspects of
22   the invention, again as most broadly conceived.   These are
23   the printer 141 (excluding its internal RIP 121N) and the
24   processor or external RIP 121E.   In addition, provision of
25   one or the other of these two units 141, 121E is an ele-
26   ment of at least one of the major method aspects of the
27   invention.
28
29   Essential to the objectives of any such system or
30   method is existence of an image 111, which may be derived
31   from a separate source and then pass through an entry
32   mechanism 112 into the computer 113 (as suggested in Fig.
33   1).   There an image is most typically subject to modifica-
34   tion in a general-purpose microprocessor 114, 119E that

1 permutation one-quarter of the time — or to use them in

2 different proportions.

3 Finally, it is possible to choose a small matrix or a

4 large one. A larger matrix will show less patterning, but

5 require more system memory. Fig. 9 provides an example of

6 how it all works together, when the above superpixel

7 definition is applied.

8

9 As noted earlier, it remains to document the 25

10 dot/mm to 25 dot/mm superpixel family (Fig. 10). It can

11 be considered an identity, and is uninteresting.

12 This time the application goes from a 25x25 dot/mm

13 cell to a 25x25 dot/mm cell. The present inventors advise

14 against use of superpixel families that average a noninte-

15 gral number of drops, as increased granularity results.

16

17

18 7.   HALFTONING

19

20 The stage that feeds superpixeling is the halftoning

21 algorithm. A preferred algorithm for use with the present

22 invention is error diffusion.

23 Error diffusion is very well known in this field. It

24 was originally conceived as a way to transform data from

25 multibit to binary (that is, single-bit). As an example

26 consider an area fill, defined at 25 dots/mm, 8 bits per

27 pixel. The whole area has the same value: tonal level

28 130 (in a conventional scale from zero through 255).

29 The only available choice is between firing a drop on

30 a given pixel location or not firing it. If the input

31 value is 0, then the system refrains from firing (0). If

32 instead the input value is 255, then the system fires (1).

33 If the input value is somewhere in between, then the

34 system goes to the closest point, but it has committed an

error; therefore it must try to commit the error in the inverse sense when moving to the neighboring pixels.

In the example, tonal value for the first pixel is 130. This is closer to 255 than to 0, so the system decides to fire (1). It has committed an error of +125, that it must then distribute among the neighbor pixels.

Assume that the next pixel receives a fourth part of the error of the previous pixel (that is, -31 counts). Then, the system must calculate that the second pixel has a value of 130 - 31 = 99. This total input value of 99 is closer to 0, so the system decides not to fire (0) — but thereby it commits an error of -99, that in turn it must propagate to the surrounding pixels (some of which will also receive error from the first pixel). This process proceeds through hundreds of thousands, or millions, of iterations to complete an image.

To fit this algorithm into the present invention, a few modifications are required. These are explored in the two subsections below.

(a) <u>Multilevel error diffusion: thresholds</u> — A first step is to conceive of a way to implement the binary outcome of classical error diffusion into a multievent (<u>i. e.</u> multibit) outcome. That is, it is no longer a binary decision between firing or not firing a drop, but rather which superpixel family to choose.

If the system is halftoning at 25 dots/mm, two bits, we'll have four superpixel families to choose among. The concept must be scalable to 12 dots/mm at four bits (sixteen superpixel families) — and even further, to six dots/mm, four bits.

Fig. 11 shows (in the two upper graphs) how the error diffusion algorithm can be expanded from binary to multi-

1 bit. At the same time, the output value has been decoup-

2 led from the actual number of drops being fired.

3   The graphs show how the contone input can be divided

4 into a number of regions equal to $2\underline{n}$ - 1, corresponding to

5 $\underline{n}$ bits per pixel at the output. Besides the two natural

6 thresholds, which are 0 and 255, new thresholds appear: A

7 and B.

8   Using this strategy, input values that are closer to

9 A generate an output to superpixel ("SPX") family 01;

10 those closer to B will be assigned to SPX 10, and so on.

11 Errors propagate in the classical way described above.

12   This explanation is the real picture for a 2 bit/pix-

13 el output, easily expanded to 4 bit/pixel or whatever is

14 required. Although Fig. 2 shows the ED thresholds A and B

15 equally spaced from 0 and 255, because of linearization

16 considerations this relationship is not maintained.

17

18   (b) <u>Linearization</u> — The classical ED algorithm was

19 originally conceived for monitor screens. On a monitor

20 screen each pixel is clearly bounded, and never overlaps

21 with the surrounding pixels. These constraints facilitate

22 good linear response of the algorithm.

23   In inkjet printing, however, the printed drops do

24 overlap. The macroscopic result is, that error diffusion

25 is no longer linear.

26   It is accordingly widely known in this field that a

27 linearization file should be created. The linearization

28 file is applied to the continuous-tone information in

29 advance of ED processing (Fig. 11, lower graphs).

30   The composite of the two functions <u>linearization</u> and

31 <u>error diffusion</u> is supposed to be the identity — so that

32 a linear contone gradient still comes out linear, once

33 halftoned. In addition, because the linearization curve

34 may assign a single image tone to different consecutive

1   inputs and thereby create contouring, the linearization

2   function also transforms the data from eight bits to nine.

3   This transformation minimizes the contouring effect.

4   The graphs also show how the intermediate thresholds

5   A, B are not evenly spaced relative to 0, 255: their

6   spacing too contributes to the linearization process.

7   Also evident is that the linearization curve is the main

8   contributor in lower-tone regions (0 to A), whereas it is

9   practically a straight line as the different thresholds

10   approach more closely (A to B, B to 255). Therefore when

11   the system halftones at 25 dots/mm at four bits, most of

12   the linearization work can be done through the threshold

13   definition.

14

15   (c)   **Linearization and threshold examples** — Finally,

16   the result for a real case in a preferred embodiment (with

17   drop table of [0 1 1 2] at 25 dots/mm, two bits) will be

18   helpful for clearer understanding (Fig. 12). This repre-

19   sents a current Hewlett Packard product.

20

21

22   8.   INK LIMITING AND PLANE SPLIT

23

24   (a)   **Overview** — Based on the foregoing understand-

25   ings of how ED works, the next step upstream in Fig. 5 is

26   to consider feeding of data into the ED. This system is

27   using plane-independent error diffusion — meaning that no

28   consideration is made, when deciding about one color, of

29   decisions already made for other colors.

30   In the product which is a preferred embodiment,

31   error-diffusion processing proceeds alternatively left to

32   right and then right to left along consecutive rows. The

33   printheads are six in number — KCMYcm — while the input

34   files are always KCMY (once they have gone through the

1   color pipeline, which may transform them from RGB to

2   KCMY).

3   In design of this system there were several choices

4   concerning the ideal point at which to split the cyan and

5   magenta planes between dark and light inks. It was de-

6   cided to split before halftoning, and thus to pass six

7   independent planes of data into the error diffusion stage.

8   The split between dark and light inks is not trivial,

9   in particular because there are different combinations of

10   dark and light ink delivering the same color, but not the

11   same total amount of ink. In other words, the plane-split

12   process must be ink-dependent.

13   Therefore, it is a good point at which to perform ink

14   limiting. The main disadvantage of this process is that

15   it operates at pixel level, not object level.

16   In other words, if there is a large solid area of the

17   same color, the system must still repeat the same opera-

18   tion for each pixel, even though it must always yield the

19   same result. This feature compels design of an algorithm

20   that gives a good tradeoff between image quality and

21   throughput.

22

23   (b) <u>Depletion algorithm</u> — We may distinguish three

24   stages in the ILPS (ink-limiting and plane-split) process

25   (Figs. 13 and 14). First, it is necessary to determine

26   how much ink is to be fired onto the particular pixel be-

27   ing processed.

28   Because of all the configurable parameters throughout

29   the halftoning pipeline (linearization, thresholds,

30   superpixel families, and drop table), it would be impossi-

31   ble to predict the ink usage based on only the values of

32   the input image. Therefore for each channel a lookup

33   table (LUT) must be built to associate the channel value

34   to the ink usage.